Minimum Color Sum of Bipartite Graphs[*]

Amotz Bar-Noy
Department of Electrical Engineering,
Tel-Aviv University, Tel-Aviv 69978, Israel.
E-mail: amotz@eng.tau.ac.il.

Guy Kortsarz
Department of Computer Science,
The Open University of Israel, Tel Aviv, Israel.
E-mail: guyk@tavor.openu.ac.il.

February 2, 1999

[*]An abstract of this paper appeared in ICALP-97.

Proposed running head: Minimum Color Sum of Bipartite Graphs.

Contact Author: Prof. Amotz Bar-Noy,

Address: Faculty of Engineering, Tel Aviv University, Tel Aviv 69978, Israel.

Phone: ++972-3-6407766

Fax : ++972-3-6407095

Email: amotz@eng.tau.ac.il

WWW: http://www.eng.tau.ac.il/ amotz/

Abstract

The problem of *minimum color sum* of a graph is to color the vertices of the graph such that the sum (average) of all assigned colors is minimum. Recently, in [BBH$^+$96], it was shown that in general graphs this problem cannot be approximated within $n^{1-\epsilon}$, for any $\epsilon > 0$, unless $NP = ZPP$. In the same paper, a 9/8-approximation algorithm was presented for bipartite graphs. The hardness question for this problem on bipartite graphs was left open.

In this paper we show that the minimum color sum problem for bipartite graphs admits no polynomial approximation scheme, unless $P = NP$. The proof is by L-reducing the problem of finding the maximum independent set in a graph whose maximum degree is four to this problem. This result indicates clearly that the minimum color sum problem is much harder than the traditional coloring problem which is trivially solvable in bipartite graphs.

As for the approximation ratio, we make a further step towards finding the precise threshold. We present a polynomial 10/9-approximation algorithm. Our algorithm uses a flow procedure in addition to the maximum independent set procedure used in previous solutions.

1 Introduction

One of the most fundamental problems in *scheduling* theory is scheduling efficiently (under some optimization goals) dependent tasks on a single machine. At any given time, the machine is capable to perform (serve) any number of tasks as long as these tasks are independent. When the serving time of each task is the same, this problem is identical to the well known coloring problem of graphs. The vertices of the graph represent the tasks and an edge in the graph between vertices v and u represents the dependency between the two corresponding tasks. That is, the machine cannot perform the tasks corresponding to vertices u and v concurrently. A similar important application arises in the context of distributed resource allocation. Here, the vertices represent processors each has one job to execute. An edge between two vertices indicates that the jobs belonging to the corresponding processors cannot be executed concurrently since they require the usage of the same common resource. This problem is known in the literature as the dining (drinking) philosophers problem ([LYN81, CM84, AS90, BP92]).

More formally, the coloring problem can be defined as follows. Let $G = (V, E)$ be an undirected simple graph with n vertices where V denotes the set of n vertices and E denotes the set of edges. A coloring of the vertices of G is a mapping into the set of positive integers, $f : V \mapsto \mathcal{Z}^+$, such that adjacent vertices are assigned different colors. We refer to $f(v)$ as the *color* of v.

The traditional optimization goal is to minimize the number of different assigned colors. We call this problem the *minimum coloring* (`MC`) problem. In the setting of tasks system, this is equivalent to finding a schedule in which the machine finishes performing all the tasks as early as possible. In the setting of resource allocation, this is equivalent to finding a schedule in which the last processor finishes executing its job the earliest. This is an optimization goal that favors the system. However, from the point of view of the tasks (or processors) themselves, we might wish to find the best coloring such that the average waiting time to be served (or to execute the job) is minimized.

Clearly, minimizing the average waiting time is equivalent to minimizing the sum of all assigned colors. The *minimum color sum* (`MCS`) problem is defined as follows. Let $G = (V, E)$ be an undirected simple graph with n vertices. We are looking for a coloring in which the sum of the assigned colors of all the vertices of G is minimized. That is, the value of $\sum_{v \in V} f(v)$ is minimized.

The minimum color sum problem was introduced by Kubicka in [K89]. In [KS89] it was shown that computing the `MCS` of a given graph is NP-hard. A polynomial time algorithm was given for the case where G is a tree. In [KKK89] it was shown that approximating the `MCS` problem within an additive constant factor is NP-hard. There, it was also shown that a first-fit algorithm yields a $(\overline{d}/2 + 1)$-approximation for graphs of average degree $\overline{d}$. Lower and upper bounds on the value of the sum coloring in general graphs were given in [TEA$^+$89].

In a recent paper, [BBH$^+$96][1], it was proven that the `MCS` problem cannot be approximated within $n^{1-\epsilon}$, for any $\epsilon > 0$, unless $NP = ZPP$. On the other hand, this paper showed that an

[1]This paper is a combination of the papers [HR93] and [BST96].

algorithm based on finding iteratively a maximum independent set is a 4-approximation to the MCS problem. This bound yields a 4ρ-approximation polynomial algorithm for the MCS problem for classes of graphs for which the maximum independent set problem can be polynomially approximated within a factor of ρ. Finally, surprisingly, in [EKS] it was shown that using optimal traditional coloring as a sub-procedure yields an unbounded approximation although coloring is "harder" than finding maximum independent set.

A special and important sub-class of graphs is the class of *bipartite graphs*. In a bipartite graph the set of vertices V is partitioned into two disjoint sets V_l and V_r such that both sets are independent. That is, all the edges of E connect two vertices one from V_l and one from V_r. Coloring V_l by 1 and V_r by 2 yields a 2-coloring of any bipartite graph. Obviously this is the best possible solution for the MC problem. However, for the MCS problem the answer is not straightforward. Denote by MBCS the MCS problem on bipartite graphs.

Coloring the largest set between V_l and V_r by 1 and the other set by 2 yields a solution to the MBCS problem the value of which is at most $3n/2$. Obviously the value of the optimal solution is at least n, and therefore this solution is at least a $3/2$-approximation to the optimal solution. The paper [BBH$^+$96] presents a better approximation of $9/8$ using as a sub-procedure the algorithm for finding a maximum independent set. In bipartite graphs, finding maximum independent set can be done in polynomial time. Therefore, their approximation algorithm is also polynomial.

New results: The contributions of this paper are the following two results:

- We prove the first hardness result for MBCS. We show that the MBCS problem admits no polynomial approximation scheme, unless $P = NP$. The proof is by L-reducing the problem of finding the maximum independent set in a graph whose maximum degree is four to the MBCS problem which implies that MBCS is MAXSNP-hard [PY88]. This result indicates clearly that the MCS problem is much harder than the traditional coloring problem.
- We improve the approximation ratio for the MBCS problem by presenting a $10/9$-approximation algorithm. Our algorithm introduces a new technique. It employs a flow procedure in addition to the maximum independent set procedure used in [BBH$^+$96].

Max-type vs. sum-type problems: Our impossibility result raises the general question of the connection between "max-type" and "sum-type" problems. The MC problem is a max-type problem whereas the MCS problem is a sum-type problem. The input and the feasible solutions for both problems are the same, the difference lies in the optimization goal. We now examine another pair of problems which relate to each other in a same manner.

The *Traveling Salesperson problem* (TSP) is defined on a set of n points with a given symmetric distance metric (d_{ij}). A feasible solution is a tour that visits each point exactly once. The traditional optimization goal is to minimize the length of the tour. Thus, the TSP problem is a max-type problem. The paper [BCC+94] deals with the *Minimum Latency Problem* (MLT). The inputs and the feasible solutions for this problem are as in the TSP problem. Let the *latency* of a point p be the length of the tour from the starting point to p. Let the total latency of the tour be the sum of latencies of all its points. The optimization goal of the MLT problem is to find a tour which minimizes the total latency. Thus, the MLT problem is a sum-type problem. Both the TSP and the MLT problems admit no bounded ratio approximation algorithm, when the distance function is arbitrary. However, both problems become easier in the metric case when the distances obey the triangle inequality. In the metric version of the problem, there exist polynomial constant-ratio approximation algorithms for both problems. The approximation ratio for the metric-TSP problem is 3/2([Chr76]). Whereas the approximation ratio for the metric-MLT problem is constant but not as small as 3/2 ([BCC+94]).

For the two coloring problems the story is different. Both the MC and the MCS problems cannot be approximated within $n^{1-\epsilon}$ for any $\epsilon > 0$ unless $NP = ZPP$ [FK96, BBH+96]. However, a big distinction exists in perhaps the easiest case of the coloring problem, namely for bipartite graphs. The remarkable property found in this paper is that, although the max-type problem, the MC problem, is trivially solvable on bipartite graphs, the sum-type problem, the MBCS problem, is MAXSNP$-$hard.

The above discussion raises the interesting question of classifying problems according to the relationship between their max-type version with the sum-type version. The coloring problem and the traveling salesperson problem each belongs to a different class.

2 Preliminaries

2.1 Notations

Given a graph $G(V, E)$ we use the following notations. For any set $S \subseteq V$, let $N(S)$ be the set of neighbors of S, i.e., the set of vertices *outside* S that are adjacent to at least one vertex of S. We also use the term S to denote the size of S.

For any graph G let $\texttt{MIS}(G)$ denote the largest independent set in G. That is, the largest subset $S \subseteq V$ such that no two vertices of S share an edge. Given a subset $X \subseteq V$ we denote by $\texttt{MIS}(X)$ the maximum independent set in the graph *induced by* X.

Given any coloring f of a graph, we denote by $\texttt{SC}(f)$ the sum of colors in f, i.e., $\texttt{SC}(f) = \sum_{v \in V} f(v)$. When $\texttt{SC}(f) = s$, we say that f has color sum s (or sum coloring s). When all the vertices in a set $S \subseteq V$ are colored by the same color c, we say that S is colored by c.

2.2 Polynomial approximation schemes

We define approximation schemes for minimization problems, a similar definition follows for maximization problems. Let P be a minimization problem. For any instance x of P, let $c_{OPT}(x)$ be the value of a minimum solution for x. We say that a polynomial algorithm $\mathcal{A}$ has approximation ratio r if for any instance x of P, algorithm $\mathcal{A}$ computes a feasible solution $\mathcal{A}(x)$ with cost $c_{\mathcal{A}}(x)$ such that:

$$\frac{c_{\mathcal{A}}(x)}{c_{OPT}(x)} \leq r \ .$$

We say that problem P admits a polynomial approximation scheme, if for any $\epsilon > 0$ there exists a polynomial time approximation algorithm for P, whose approximation ratio is bounded by $(1+\epsilon)$.

2.3 $L-$reduction

The L-reduction ([PY88]) is a tool that helps proving hardness results. Unlike the usual NP-hardness reductions, it "preserves" approximation ratios (in a sense to be described). Therefore, it can be used in showing that a given problem admits no polynomial approximation scheme.

In order to define L-reduction we need the following notations. Let P be an optimization (either minimization or maximization) problem. Denote by $I(P)$ the set of instances for problem P, by $sol(P)$ the set of feasible solutions of problem P, and by $c_P(s)$ the cost function of any feasible solution s for P.

Suppose now that P and Q are two optimization problems. In order to construct an $L-$reduction we need to define two (polynomially computable) functions $\mathcal{R} : I(P) \mapsto I(Q)$ and $\mathcal{S} : sol(Q) \mapsto sol(P)$. For any instance $x \in I(P)$ let $c_{OPT}(x)$ be the value of the optimal solution for x and let $c_{OPT}(\mathcal{R}(x))$ be the value of the optimal solution for $\mathcal{R}(x)$. The two functions $\mathcal{R}$ and $\mathcal{S}$ are an $L-$reduction from problem P to problem Q, if there exist two constants α and β such that the two following properties hold:

1. $c_{OPT}(\mathcal{R}(x)) \leq \alpha \cdot c_{OPT}(x)$.
2. For any feasible solution $s \in sol(Q)$ of $\mathcal{R}(x)$, $\mathcal{S}(s)$ is a feasible solution for x and

$$|c_{OPT}(x) - c_P(S(s))| \leq \beta \cdot |c_{OPT}(\mathcal{R}(x)) - c_Q(s)| \ .$$

The following theorem is shown in [PY88].

Theorem 2.1 *Suppose that Problem P admits no polynomial approximation scheme and that Problem P can be $L-$reduced to problem Q. Then Problem Q admits no polynomial approximation scheme.* ▮

2.4 The `MIS` and `4-MIS` problems

The *Maximum Independent Set* (`MIS`) problem is the following. Given an undirected graph $G(V, E)$ with n vertices, the goal is to find a maximum independent set. I.e., a maximum sized set $S \subseteq V$ such that no two vertices of S share an edge. In a recent paper ([Has96]), it was shown that, unless $P = NP$, the `MIS` problem has no n^{ϵ}-approximation algorithm for any fixed $0 < \epsilon < 1$.

The Δ-`MIS` problem is the `MIS` problem restricted to graphs with maximum degree Δ. For this problem there exists a simple greedy algorithm with approximation ratio $(\Delta + 1)$. In any iteration, pick a vertex v not yet removed, add it to S, and remove v and its neighbors from the graph. This greedy algorithm also indicates that a graph of maximum degree Δ always contains an independent set of size at least $n/(\Delta+1)$. In fact, it was shown by Turan [T41] and Erdös [E70], that the greedy algorithm produces an independent set of size at least $n/(\delta + 1)$, where δ is the *average* degree of the graph. In [HR94] it is shown that the approximation ratio of the greedy algorithm is in fact $(\Delta + 2)/3$.

The first approximation algorithm for the Δ-`MIS` problem, that extended and improved the greedy algorithm, is due to Hochbaum [H83] and has $(\Delta + 1)/2$ approximation ratio. Better approximation algorithms for the Δ-`MIS` problem were shown in [BF94, HR94]. The best currently known algorithm for this problem has approximation ratio roughly $\Delta/6$ ([HR94]).

We need the following theorem from [ALM$^+$92].

Theorem 2.2 *There exists some $\epsilon > 0$ such that the* `4-MIS` *admits no $(1 + \epsilon)$-approximation algorithm, unless $P = NP$ (and hence* `4-MIS` *admits no polynomial approximation scheme).* ▮

Remark: This result is true for any `MAXSNP`−hard (or complete) problem such as vertex cover, max-2sat, and max-cut. (see Theorem 2.1).

2.5 Known algorithms for the `MBCS` problem

We recall the approximation algorithm presented in [BBH$^+$96]. For a given bipartite graph G, denote by I_1 the maximum independent set in G, by I_2 the maximum independent set in $G \setminus I_1$, by I_3 the maximum independent set in $G \setminus (I_1 \cup I_2)$, and so on. The algorithm of [BBH$^+$96] is best explained by the definition of a sequence of (roughly) $\log n$ possible algorithms.

Let `A(2)` be the algorithm that colors the vertices of G with two colors, the larger side of V by 1 and the smaller side by 2. Let `A(3)` be the following algorithm: color the vertices of I_1 by 1, and then color the vertices of $G \setminus I_1$ by 2 and 3 (i.e., color the larger side in the remaining graph by 2 and the smaller side by 3). In general, for $i \geq 3$ and for $1 \leq j \leq i - 2$, algorithm `A(i)` colors the sets I_j with color j, and then colors the larger side of the remaining graph by $i - 1$ and the smaller side by i. All together, algorithm `A(i)` uses i colors. Note that we have defined at most $\lfloor \log n \rfloor$ algorithms, because the maximum independent set in any bipartite graph with n vertices contains

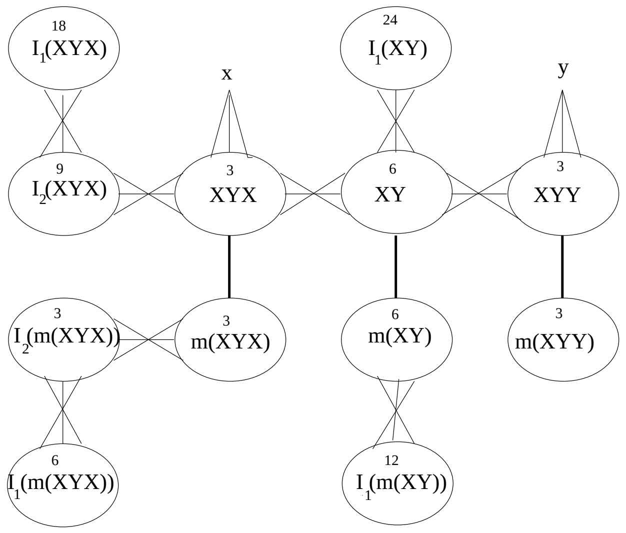

Figure 1: The sets of vertices and edges in the gadget corresponding to the edge (x, y).

at least $n/2$ vertices. Let A' be the last possible algorithm in this family of algorithms.

Since G is a bipartite graph, it follows that $I_1 \geq n/2$. Therefore, algorithm `A(2)` is a 3/2-approximation algorithm. Consider now the following algorithm, denoted by $\mathcal{B}$, that runs algorithms `A(2)` and `A(3)` and picks the best solution. The following theorem is proved in [BBH$^+$96].

Theorem 2.3 *Algorithm $\mathcal{B}$ is a $9/8$-approximation algorithm to the* `MBCS` *problem.* ▮

Remark: We can prove some further results (details are omitted). Algorithm A' (when taken alone) has an approximation ratio 4/3. Also, it does not help to pick the best of the first i algorithms, since it is possible to show that the 9/8 ratio still holds. Thus some new ideas are in order.

3 A hardness result for the `MBCS` problem

In this section, we prove that (unless $P = NP$) the `MBCS` problem has no polynomial approximation scheme. We do that by proving an $L-$reduction from the `4-MIS` problem to the `MBCS` problem (hence showing that the `MBCS` problem is `MAXSNP`$-$hard). By Theorems 2.1 and 2.2 the hardness result is implied.

3.1 The construction – the function $\mathcal{R}$

Let $G(V, E)$ be an instance of the `4-MIS` problem. Thus, the maximum degree in G is bounded by 4. The $\mathcal{R}$ function should map G into a graph $\tilde{G}$ which is an instance of the `MBCS` problem. In this subsection we describe $\tilde{G}$.

The graph $\tilde{G}$ contains a vertex corresponding to each vertex in V and these vertices form an independent set. We assume an order on the vertices of G. Whenever we consider an edge $(x, y) \in E$ we assume that $x < y$. The construction involves adding a *gadget* for each edge $e = (x, y) \in E$. Each gadget is composed of twelve independent sets of vertices containing no internal edges (edges only cross from one different set to the other). The sets of vertices corresponding to different edges are *disjoint*.

Before describing the sets of vertices and the edges of any gadget we need some definitions. We say that two (independent) sets A and B are *cliqued*, if every vertex in A is connected to every vertex in B that is, the sets A and B induce a complete bipartite graph. We say that the two sets are *matched* if $|A| = |B|$ and every vertex x in A has a *single* neighbor $m(x)$ in B, that is, the sets A and B induce a perfect matching. The sets and edges in the gadget corresponding to the edge $e = (x, y)$ are as follows.

Main and matched sets:

1. A set XYX of 3 vertices and a *matched* set $m(XYX)$ of 3 vertices.
   The sets XYX and $m(XYX)$ are matched.

2. A set XYY of 3 vertices and a matched set $m(XYY)$ of 3 vertices.
   The sets XYY and $m(XYY)$ are matched.

3. A set XY of 6 vertices and a matched set $m(XY)$ of 6 vertices.
   The sets XY and $m(XY)$ are matched.

Imposing sets:

1. A set $I_1(XYX)$ of 18 vertices and a set $I_2(XYX)$ of 9 vertices.
   The two sets $I_1(XYX)$ and $I_2(XYX)$ are cliqued.

2. A set $I_1(m(XYX))$ of 6 vertices and a set $I_2(m(XYX))$ of 3 vertices.
   The sets $I_1(m(XYX))$ and $I_2(m(XYX))$ are cliqued.

3. Two sets $I_1(XY)$ of 24 vertices and $I_1(m(XY))$ of 12 vertices.

Additional edges between the sets:

1. The vertex x is connected to all 3 vertices of XYX.
   The vertex y is connected to all 3 vertices of XYY.

2. The sets XYX and XY are cliqued.
   The sets of XYY and XY are cliqued.

3. The sets XYX and $I_2(XYX)$ are cliqued.
   The sets $m(XYX)$ and $I_2(m(XYX))$ are cliqued.

4. The sets XY and $I_1(XY)$ are cliqued.
   The sets $m(XY)$ and $I_1(m(XY))$ are cliqued.

This completes the description of the gadget corresponding to each edge $e = (x, y)$ and the description of the $\mathcal{R}-$function. The above sets depend on e, that is, there is such a gadget for every edge $e \in E$. We avoid adding e as a subscript in these sets, for the simplicity of notation. In order for the $\mathcal{R}$ function to be valid we demonstrate a 2 coloring for $\tilde{G}$ proving that the graph $\tilde{G}$ is a bipartite graph.

Claim 3.1 *The graph $\tilde{G}$ is bipartite.*

Proof: Color the independent set corresponding to V, and the six sets $m(XYX)$, $m(XYY)$, XY, $I_1(m(XYX))$, $I_1(m(XY))$ and $I_2(XYX)$ by 1. Color the rest of the vertices in $\tilde{G}$ by 2. Since all the edges defined above connect vertices colored by 1 with vertices colored by 2, it follows that this is a legal $2-$coloring for $\tilde{G}$. ▮

3.2 The intuition behind the construction

The goal of the construction is to enable us to define the right function $\mathcal{S}$. This will be explain in the next subsection. Here we give some intuition.

The role of the imposing sets is to force a situation in which some sets cannot be colored by a specific color. For example, it will be shown that in an optimal coloring the imposing set $I_2(XYX)$ is colored by 2. Consequently, the set XYX cannot be colored by 2. In general, in an optimal solution, all the sets of type I_1 are colored by 1 and all the sets of type I_2 are colored by 2.

The role of the matched sets is to assure that the sum coloring of two matched sets is fixed in any optimal coloring. For example, if a vertex in XYX is colored by 1, then its matched vertex is colored by 3, and vice versa (recalling that these two sets can not be colored by 2 because of the two imposing sets $I_2(XYX)$ and $I_2(m(XYX))$). Thus every pair in XYX and $m(XYX)$ adds exactly 4 to the sum coloring in an optimal coloring and the contribution of XYX and $m(XYX)$ is fixed.

Now let us explain the main idea in the construction. Let x and y be two vertices adjacent in G (i.e., $(x, y) \in E$). We will show that we lose in the sum coloring if both x and y are colored by 1. Indeed, say that both x and y are colored 1, and consider the colors of XY, XYX, XYY. In the best coloring XYX is colored by 3 and XYY by 2. Therefore, since the set $I_1(XY)$ is colored by 1, it follows that XY is colored by at least 4. On the other hand, if one of x and y is not colored by 1, we may gain by assigning XY a color less then 4. This follows since XYX and XYY will "waste" only one of the colors 2 and 3. Hence, it is possible to color XY with either 2 or 3.

Therefore, a "good" sum coloring would color by 1 an *independent set* in G. In addition, a "good" sum coloring would strive to color as many vertices of G as possible by 1. It therefore pays to color as large as possible independent set in G by 1. Thus, a "good" approximation for the `MBCS` problem implies a "good" approximation for the `4-MIS` problem.

3.3 The function $\mathcal{S}$

We need the following definition for the construction of $\mathcal{S}$. A coloring $\tilde{f}$ of the vertices in $\tilde{G}$ is *proper*, if the two following properties hold for every edge.

Imposing properties:

1. The sets $I_1(XYX)$, $I_1(m(XYX))$, $I_1(XY)$, and $I_1(m(XY))$ are colored by 1.
2. The sets $I_2(XYX)$ and $I_2(m(XYX))$ are colored by 2.

Independence property:

All the vertices of G that are colored by 1 in $\tilde{f}$ form an independent set in G.

The process of constructing $\mathcal{S}$ is as follows. We start with any feasible coloring f of $\tilde{G}$. We then show in five stages that f can be transformed to a *proper* coloring $\tilde{f}$ such that the sum of colors in $\tilde{f}$ is no larger than the sum of colors in f ($\texttt{SC}(\tilde{f}) \leq \texttt{SC}(f)$). The mapping $\mathcal{S}$ is now defined by choosing the set of vertices in G that are colored by 1 by $\tilde{f}$ denoted by $I_1(\tilde{f})$. Note, that by the independence property, $I_1(\tilde{f})$ is also an independent set in G.

In the first stage we transform f into f_1 such that all the vertices in any independent set in any gadget are colored by the same color. In the second stage, we transform f_1 into a "locally minimal" coloring f_2. That is a coloring in which each set in the gadget is colored by no more than $k+1$ where k is the number of neighboring sets to this set. In the third stage, we show how to transform f_2 into a coloring f_3 such that the imposing properties hold. In the forth stage, we transform f_3 into a coloring f_4 in which all the sets XYX and XYY in all the gadgets are colored by no more than 3. Finally, in the fifth stage we transform f_3 into the desired coloring $\tilde{f}$ by showing how to achieve the independence property. In all five stages the new coloring has no worse sum coloring then the previous one. Fix an edge $e = (x, y)$, the five stages are stated in lemmas 3.2, 3.3, 3.4, 3.5, and 3.6.

Lemma 3.2 *Let f be a legal coloring of $\tilde{G}$. Then there exists a coloring f_1 of $\tilde{G}$ such that*

1. *All the vertices in any set in the gadget are colored by the same color.*
2. $\texttt{SC}(f_1) \leq \texttt{SC}(f)$.
3. *The vertices corresponding to the vertices of G are colored the same in both f and f_1.*

Proof: Let A be one of the imposing sets in the gadget. Let c be the minimum color of any vertex in the set A. Color all the vertices in A by c. Since all the vertices in A are connected in the same fashion to the vertices outside of A, it follows that this coloring is legal. Let A and B be two matched sets in the gadget. Let $u \in A$ and $v \in B$ be two matched vertices such that the sum of their colors is minimal. Color all the vertices of A by the color of u and all the vertices of B by the color of v. Since any pair of matched vertices in A and B are connected in the same fashion to the vertices outside A and B, it follows that this coloring is legal. Thus the first property holds. The second property follows since we did not increase the sum coloring of any imposing set and any pair of matched sets. The third property follows since we did not touch the vertices of G. ▮

Lemma 3.3 *Let f_1 be the coloring of $\tilde{G}$ constructed from f as implied by Lemma 3.2. Let A be one of the sets in the gadget. Let k be the number of neighboring sets of A. (For that purpose, we consider x and y as sets of size 1. For example, if $A = XYX$ then $k = 4$). Then there exists a coloring f_2 of $\tilde{G}$ such that*

1. *The color of A is at most $k + 1$.*
2. $\mathtt{SC}(f_2) \leq \mathtt{SC}(f)$.
3. *All the properties of f_1 remain.*

Proof: The neighboring sets of A can occupy at most k different colors. Hence, one color less than or equal $k+1$ is legal for A. If A is colored by a color larger than $k+1$, re-color it by this free color. Thus the first property holds. The second property follows since we did not increase the sum coloring of any set. The third property follows since we did not touch the vertices of G. ▮

Lemma 3.4 *Let f_2 be the coloring of $\tilde{G}$ constructed from f as implied by Lemma 3.2 and Lemma 3.3. Then there exists a coloring f_3 of $\tilde{G}$ such that*

1. *The imposing properties hold for f_3.*
2. $\mathtt{SC}(f_3) \leq \mathtt{SC}(f)$.
3. *All the properties of f_2 remain.*

Proof: We first show how to color the I_1-type sets by 1 without increasing the sum coloring. By Lemma 3.3, we get that the color of $I_2(XYX)$ is at most 3 and the color of $I_1(XYX)$ is at most 2. If $I_1(XYX)$ is not colored by 1, then re-color it by 1. In case $I_2(XYX)$ was colored by 1, re-color it by the smallest legal color. This smallest color is at most 3. This results in a legal coloring in which $I_1(XYX)$ is colored by 1. Since $|I_1(XYX)| \geq 2|I_2(XYX)|$ and since we gained $|I_1(XYX)|$ and lost at most $2|I_2(XYX)|$, it follows that the new coloring has a sum coloring which is no worse than the previous sum coloring. Similar reasoning shows how to color the sets $I_1(m(XYX))$, $I_1(XY)$, and $I_1(m(XY))$ by 1.

We now show how to color the I_2-type sets by 2 without increasing the sum coloring. If $I_2(XYX)$ is not colored by 2 then it is colored by 3 (by Lemma 3.3). If this is the case, re-color $I_2(XYX)$ by 2. As a consequence, we might need to change the color of XYX from 2 to 5. Since $|I_2(XYX)| \geq 3|XYX|$, it follows that the new coloring has a sum coloring which is no worse than the previous sum coloring. By similar reasoning, we can re-color $I_2(m(XYX))$ by 2 if it is not colored by 2. This is because $|I_2(m(XYX))| \geq |m(XYX)|$ and re-coloring $I_2(m(XYX))$ by 2 would increase the color of $m(XYX)$ from 2 to 3 at most.

We use the above two transformations to get a coloring f_3 for which the imposing properties hold without increasing the sum coloring and without changing the color of any vertex in G. ▮

Lemma 3.5 *Let f_3 be the coloring of $\tilde{G}$ constructed from f as implied by Lemma 3.2, Lemma 3.3, and Lemma 3.4. Then there exists a coloring f_4 of $\tilde{G}$ such that*

1. *The sets XYX and XYY in are colored by at most* 3.
2. $\texttt{SC}(f_4) \leq \texttt{SC}(f)$.
3. *The set of vertices colored* 1 f_4 *is a subset of the vertices colored* 1 *in* f_3.

Proof: Assume that there exist sets XYX or XYY that are colored by 4 or more. Re-color (simultaneously in all gadgets) all these XYX and XYY sets by 1, the corresponding $m(XYX)$ sets by 3, and the corresponding $m(XYY)$ sets by 2. Note that the color of XYX and XYY does not conflict with the color of XY: XY is not colored by 1 because of $I_1(XY)$. Now, re-color by 4 any vertex x or y that is colored by 1 such that its corresponding set XYX or XYY is colored by 1 as well. We claim that the sum coloring of the new coloring is no more than the sum coloring of the previous coloring. This follows since for any vertex x that was re-colored from 1 to 4, its corresponding XYX was re-colored from 4 to 1, and $m(XYX)$ from (at least) 1 to 3. Thus we gain $3XYX = 9$ and lose at most $2m(XYX) + 3 = 9$. The analysis for a vertex y is similar. ▮

We are now ready to describe the fifth stage. For a coloring g, Let $I_1(g)$ be the set of vertices in G colored by 1 in g.

Lemma 3.6 *Let f_4 be the coloring of $\tilde{G}$ constructed from f as implied by Lemma 3.2, Lemma 3.3, Lemma 3.4, Lemma 3.5 and Lemma 3.6. Then there exists a coloring $\tilde{f}$ of $\tilde{G}$ such that*

1. *The independence property holds for $\tilde{f}$. Moreover, if $I_1(\tilde{f})$ is the set of vertices in G colored by 1 by $\tilde{f}$, then $I_1(\tilde{f}) \subseteq I_1(f)$.*
2. *The imposing property holds for $\tilde{f}$.*
3. $\texttt{SC}(\tilde{f}) \leq \texttt{SC}(f)$.

Proof: For the independence property, we need to change colors so that no two vertices x and y that are adjacent in G are colored by 1. Recall that by Lemma 3.2 all the vertices in any set are colored by the same color and that this color is locally minimal by Lemma 3.3 (that is the color of each set is no more than $k+1$, if the set have k neighboring sets). We perform the following changes (iteratively) for every pair of vertices x and y that are colored by 1 and are adjacent in G.

First note that XYX is colored by 3. This follows since XYX is not colored by 1 due to x, is not colored by 2 due to $I_2(XYX)$, and is not colored by 4 or more due to Lemma 3.5. We now show how to color XYY by 2, without increasing the sum coloring. Supposed that XYY is not colored by 2. Re-color XYY by 2, $m(XYY)$ by 1, XY by 3, $m(XY)$ by 2, XYX by 1, $m(XYX)$ by 3, and x by 4. Note that we gain at least 3 in the sum coloring for the re-coloring of the vertices in XYY and lose only 3 for re-coloring x. Thus, x is not colored by 1 anymore. Assume now that all the vertices in XYY are colored by 2. It is now necessarily the case that XY is colored by at least 4. This is because XY is not colored by 1 due to $I_1(XY)$, is not colored by 3 due to XYX, and is not colored by 2 due to XYY. Our final re-coloring is as follows. We re-color XY by 3, $m(XY)$ by 2, XYX by 1, $m(XYX)$ by 3, XYY by 1, $m(XYY)$ by 2, and both x and y by 4. We gain at least 6 for the re-coloring of the vertices in XY and lose at most 6 for the re-coloring of x and y.

In the transformations described above we did not increase the sum coloring. Moreover, the only changes in the colors of vertices in G are from color 1 to color 4 proving the first claim of the lemma. ▮

The function $\mathcal{S}$ on any legal coloring f of $\tilde{G}$ is defined as follows. Let $\tilde{f}$ be the proper coloring constructed from f as implied by Lemmas 3.2, 3.3, 3.4, 3.5, and 3.6. Let $I_1(\tilde{f})$ be the set of vertices colored by 1 in $\tilde{f}$. By Lemma 3.5, this is a feasible independent set. Then

$$\mathcal{S}(f) = I_1(\tilde{f})$$

3.4 The $L-$reduction properties

We now turn to prove the two $L-$reduction properties. Let $\mathtt{OPT}$ be the minimum sum coloring in $\tilde{G}$ and let $\mathtt{MIC} = \mathtt{SC}(\mathtt{OPT})$. The next lemma proves the first property of the L-reduction.

Lemma 3.7 *There exists a constant* α *such that* $\mathtt{MIC} \leq \alpha \cdot \mathtt{MIS}(G)$.

Proof: First note that the degrees in the graph G are at most 4. Consequently, $\mathtt{MIS}(G) \geq n/5$. Also note that $\tilde{G}$ has $O(n)$ vertices. This is because G has $O(n)$ edges, and $\tilde{G}$ has $O(1)$ additional vertices per any edge in G. Now since $\tilde{G}$ is a bipartite graph and therefore can be colored by 1 and 2, it follows that $\mathtt{MIC} = O(n)$. These two facts imply the first property. ▮

For the second property of the L-reduction, we need to show the existence of a constant β such that for any legal coloring f of $\tilde{G}$ the following holds: $\mathtt{MIS}(G) - \mathcal{S}(f) \leq \beta(\mathtt{SC}(f) - \mathtt{MIC})$. We prove this inequality with $\beta = 1$. The proof uses the following two claims. Let I_1 be the maximum independent set in G.

Claim 3.8 $\mathtt{MIC} \leq 135 \cdot E + 2n - I_1$.

Proof: Color I_1 by 1 and the rest of the vertices in G by 2. Let $(x, y) \in E$, note that x and y are not both colored by 1 since I_1 is an independent set. We first color the imposing sets of type I_1 by 1 and the imposing sets of type I_2 by 2. The contribution of the imposing sets to the sum coloring per edge is $18 \cdot 1 + 9 \cdot 2 + 6 \cdot 1 + 3 \cdot 2 + 24 \cdot 1 + 12 \cdot 1 = 84$. Now consider the following three possible cases.

1. Vertex x is colored by 1 and vertex y is colored by 2. Color XYX by 3, $m(XYX)$ by 1, XYY by 1, $m(XYY)$ by 2, XY by 2, and $m(XY)$ by 3.
2. Vertex x is colored by 2 and vertex y is colored by 1. Color XYX by 1, $m(XYX)$ by 3, XYY by 2, $m(XYY)$ by 1, XY by 3, and $m(XY)$ by 2.
3. Both vertices x and y are colored by 2. Color XYX by 3, $m(XYX)$ by 1, XYY by 1, $m(XYY)$ by 2, XY by 2, and $m(XY)$ by 3.

The contribution of all the matched sets to the sum coloring in all three cases is $3(1 + 3) + 3(1 + 2) + 6(2 + 3) = 51$. All together, each gadget in $\tilde{G}$ contributes 135 to the sum coloring. Since the vertices of $\tilde{G}$ contribute $2n - I_1$ to the sum, the claim follows. ▮

Now let f be an arbitrary coloring of $\tilde{G}$ and let $\tilde{f}$ be its corresponding proper coloring. Let $I_1(\tilde{f})$ be the set of vertices colored by 1 in $\tilde{f}$, and thus $\mathcal{S}(f) = I_1(\tilde{f})$.

Claim 3.9 $\texttt{SC}(\tilde{f}) \geq 135 \cdot E + 2n - I_1(\tilde{f})$.

Proof: Since $\tilde{f}$ is a proper coloring, the contribution of the imposing sets to the sum coloring per edge is 84 as was shown in the previous claim. Now fix an edge and consider the three pairs of matched sets.

1. The sets XYX and $m(XYX)$ contribute at least $3 \cdot (1 + 3)$ to the sum coloring. This is because both sets can not be colored by 2.
2. The sets XYY and $m(XYY)$ contribute at least $3 \cdot (1 + 2)$ to the sum coloring. This is because one set must be colored by 2.
3. The sets XY and $m(XY)$ contribute at least $6 \cdot (2 + 3)$ to the sum coloring. This is because both sets cannot be colored by 1.

All together the contribution of the matched sets to the sum coloring per edge is at least 51. The lower bound derived so far is $135 \cdot E$. The claim follows since the set $G \setminus I_1(\tilde{f})$ contributes at least $2n - 2I_1(\tilde{f})$ to the sum coloring. ▮

Lemma 3.10 $\texttt{MIS}(G) - \mathcal{S}(f) \leq \texttt{SC}(f) - \texttt{MIC}$.

Proof: The following inequalities are implied by Lemma 3.6 Claim 3.8, and Claim 3.9.

$$\begin{aligned}
\mathtt{SC}(f) - \mathtt{MIC} &\geq \mathtt{SC}(\tilde{f}) - \mathtt{MIC} \\
&\geq (135 \cdot E + 2n - I_1(\tilde{f})) - (135 \cdot E + 2n - I_1) \\
&= \mathtt{MIS}(G) - \mathcal{S}(f) \ .
\end{aligned}$$

▮

We completed constructing a valid L-reduction from the 4-MIS problem to the MBCS problem. The following theorem follows from Theorems 2.1 and 2.2.

Theorem 3.11 *There exists an $\epsilon > 0$ such that there is no $(1+\epsilon)-$ratio approximation algorithm for the* MBCS *problem unless $P = NP$.* ▮

4 Improved approximation algorithm for MBCS

In the previous section we have shown that there exists some $\epsilon > 0$ such that the MBCS problem has no $(1+\epsilon)$-approximation algorithm. However, the precise threshold for the approximation is yet to be determined. We take a further step in this direction. In this section, we present a new algorithm $\mathcal{C}$ that utilizes a new procedure Neig. We prove that this procedure, combined with algorithms A(2), A(3), and A(4) (see section 2.5) yield a 10/9-approximation algorithm for the MBCS problem. This improves the previous 9/8-approximation algorithm of [BBH$^+$96].

4.1 Some intuition

In this subsection, we reveal some of the intuition behind our algorithms. For that end, we relay on Figure 2. Indeed, let I_1 be a maximum independent set in G, and let I_l^1 and I_r^1 be its respective sides. Let Z (resp. W) be the larger (resp. smaller) of the two sides in $V \setminus I_1$.

In the $9/8-$ratio approximation algorithm of [BBH$^+$96] the following lower bound was used for the optimum sum coloring. At best, the optimum algorithm can pick a "good" maximum independent set I_1^* in G, so that *all* the remaining vertices of the graph will form an independent set $I_2^* = V \setminus I_1$. This gives a $2 \cdot n - I_1^* = 2 \cdot n - I_1$ lower bound for the optimum sum.

Let I_2 be the maximum independent set in $Z \cup W$. Our aim is to prove that either $(Z+W) - I_2$ is "large", or one of the algorithms A(2) or A(3) or A(4) has approximation ratio better than 9/8. This is done as follows.

1. We may assume that (the size of) Z roughly equals (the size of) W, for otherwise, A(3) has good ratio (for example, if $W = \emptyset$, then A(3) is optimal.)

2. Next, we may assume that I_2, the maximum independent set in $W \cup Z$, is not very large. For example, if $I_2 = Z \cup W$, clearly A(3) is again optimal. Even if I_2 is *slightly* smaller than

$Z \cup W$, coloring G with 4 colors using Algorithm `A(4)`, gives a good approximation, because the sets colored 3 and 4 are very small.

3. Finally, we may assume that W is not too small (and therefore Z is not very small either). To understand that consider the case where W and Z are almost empty. Clearly, in this case we have colored almost all the vertices with color 1, thus resulting in a "good" coloring.

Note by 2 and 3 above, that we get that $(Z+W)-I_2$ is large, as required. This means that there is no "large" independent set in the graph induced by $Z \cup W$. Now, if the optimum wants to match the lower bound of $2 \cdot n - I_1$, this means that many vertices of $Z \cup W$ should be colored 1 in the optimum coloring. Thus, the main idea is to define a new procedure, `Neig`, that starts with the coloring of `A(3)`, using some arbitrary maximum independent set I_1 with some corresponding Z and W, and tries to find a good candidate subset $S \subseteq Z \cup W$, to be recolored by 1. Note that automatically, all the neighbors of S need to be colored by some color different than 1.

If such a good set S does not exist, it means that we can increase (and therefore improve) the lower bound $2 \cdot n - I_1$ for the optimum coloring. The main difficulty hence, is choosing a good S. Although we are not able to compute the best set S to be recolored by 1, we can find a good "half" of S. Namely, we can find the best set $S_W \subseteq W$, or $S_Z \subseteq Z$ to be recolored 1, so as to reduce the sum as much as possible. Thus, we do not choose the best S to move from colors 2 and 3 to color 1, but we can choose a subset S_W or S_Z, that reduces the sum by at least "half" of the reduction in the best choice of S.

4.2 An algorithmic tool

We now describe the new tool used in our approximation algorithm. Define the `2-Neighborhood` problem as follows. Given a bipartite graph $G(V_l, V_r, E)$ we look for a set $S_l \subseteq V_l$ such that $d_{S_l} = 2S_l - N(S_l)$ is maximum. Recall that $N(S_l)$ is the set of vertices outside S_l that have at least one neighbor in S_l. We note that the order in which V_l and V_r are specified in the problem-presentation is important, that is the solution S_l is a subset of V_l.

A generalization of this problem, called *the selection problem* was first discussed by Rhys and Balinski. Note that this problem is also called "a provision problem" in Chapter 5 of [L76]. In the selection problem, the goal is to maximize the following objective:

$$\sum a_i x_i - \sum b_i y_i,$$

where x_i is 1, only if the respective vertex in the left side is included, and y_i is 1 only if the respective vertex on the right side in included. The restriction in the problem is that a vertex on the left is included, only if all its right-side neighbors are included as well. Thus, it is not hard to see this problem can be solved via flow methods (combined with binary search procedure). The currently fastest algorithm for the selection problem is given in [GGT89]. For the sake of completeness, we

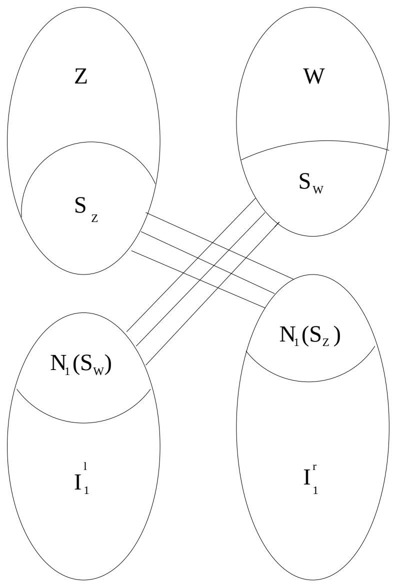

Figure 2: The sets in the bipartite graph G used by Procedure `Neig`.

give a brief description of the flow algorithm that solves our special case of the selection problem, the `2-Neighborhood` problem.

We construct the following directed bipartite graph with capacities, denoted by $G' = (V', E')$. The set of vertices V' contains V and two additional vertices: a source s and a sink t. The set of edges E' contains all the edges in E directed from V_l to V_r with capacity ∞. In addition, E' contain V_l edges with capacity 1 emanating from s to all the vertices of V_l and V_r edges with capacity $1/2$ emanating from all the vertices in V_r to t.

We now show how to deduce the solution for the `2-Neighborhood` problem from a minimum cut in G'. Let V_l^s and V_r^s be the subsets of V_l and V_r that are with s in the same side of the minimum cut. Let V_l^t and V_r^t be the complementing sets. Note that the cut containing s alone in one side is finite. Therefore, any vertex in V_l^s is not connected to any vertex in V_r^t, since otherwise the capacity of the cut is ∞. Thus, the capacity of the cut is given by the following expression: $V_l^t + V_r^s/2$. The chosen cut minimizes this expression, and therefore maximizes the following expression:

$$2\left(V_l - V_l^t - \frac{V_r^s}{2}\right) = 2V_l^s - V_r^s \ .$$

Since the capacity of the minimum cut is finite, it follows that no vertex in V_r^t has a neighbor in V_l^s. Moreover, every vertex in V_r^s has at least one neighbor in V_l^s, because otherwise moving this vertex to V_r^t would decrease the capacity of the cut. Therefore, $V_r^s = N(V_l^s)$ which implies that V_l^s is the solution for the `2-Neighborhood` problem.

4.3 Procedure `Neig` and Algorithm $\mathcal{C}$

Procedure `Neig` utilizes the solution to the `2-Neighborhood` problem described in section 4.2. We now define subsets of the vertices of the graph G and subgraphs of G used by procedure `Neig`.

1. I_1 – the maximum independent set in G.
   $I_1^l = I_1 \cap V_l$ – the left side of I_1
   $I_1^r = I_1 \cap V_r$ – the right side of I_1.

2. Z – the larger side of $G \setminus I_1$.
   W – the smaller side of $G \setminus I_1$.
   Without loss of generality, assume that $Z \subset V_l$ and $W \subset V_r$.

3. $G_Z = (Z, I_1^r, E_Z)$ – the (bipartite) subgraph induced by Z and I_1^r.
   $G_W = (W, I_1^l, E_W)$ – the (bipartite) subgraph induced by W and I_1^l.

4. S_Z – the set maximizing $d_{S_Z} = 2S_Z - N(S_Z)$ in G_Z.
   S_W – the set maximizing $d_{S_W} = 2S_W - N(S_W)$ in G_W.
   Note that W plays here the role of the left side of the `2-Neighborhood` problem.

5. $N_1(S_Z) = N(S_Z) \cap I_1^r$ – the neighbors of S_Z in I_1^r.
   $N_1(S_W) = N(S_W) \cap I_1^l$ – the neighbors of S_W in I_1^l.
   Note that $N(S_Z)$ ($N(S_W)$) in G can contain vertices from W (Z). Therefore, we need to define $N_1(S_Z)$ ($N_1(S_W)$).

Procedure `Neig`:

If $d_{S_Z} \geq d_{S_W}$:
1. Color $I_1^l \cup S_Z \cup (I_1^r \setminus N_1(S_Z))$ by 1.
2. Color $W \cup N_1(S_Z)$ by 2.
3. Color $Z \setminus S_Z$ by 3.

If $d_{S_Z} < d_{S_W}$:
1. Color $I_1^r \cup S_W \cup (I_1^l \setminus N_1(S_W))$ by 1.
2. Color $Z \cup N_1(S_W)$ by 2.
3. Color $W \setminus S_W$ by 3.

For the case $d_{S_Z} \geq d_{S_W}$, procedure `Neig` can be described as follows. Start with the initial coloring of `A(3)`, i.e., I_1 is colored by 1, Z (the larger of the two remaining sides) is colored by 2, and W is colored by 3. Thus $\texttt{SC(A(3))} = I_1^l + I_1^r + 2Z + 3W$. Next, re-color Z by 3 and W by 2, loosing $Z - W$ in the sum coloring. Next, change the color of S_Z from 3 to 1 gaining $2S_Z$ in the sum coloring. This forces all the neighbors of Z in I_1, $N_1(S_Z)$, to be colored by a color different than 1, thus color them by 2. Here we lose $N_1(S_Z)$ in the sum coloring. The net profit in the sum coloring is therefore $2S_Z - N_1(S_Z) + W - Z = d_{S_Z} + W - Z$. Similarly, it can be shown that for the case $d_{S_W} > d_{S_Z}$, the net profit is d_{S_W}. (This case is better for us since we do not need to switch the colors of Z and W, loosing $Z - W$.) Thus, we proved the following proposition.

Proposition 4.1

(1). If $d_{S_Z} \geq d_{S_W}$ then $\texttt{SC(Neig)} = \texttt{SC(A(3))} - d_{S_Z} + (Z - W)$.

(2). If $d_{S_W} > d_{S_Z}$ then $\texttt{SC(Neig)} = \texttt{SC(A(3))} - d_{S_W}$.

We conclude this subsection with the description of algorithm $\mathcal{C}$. It clearly follows that the algorithm has a polynomial running time.

Algorithm $\mathcal{C}$

- Run algorithms `A(2)`, `A(3)`, `A(4)`, and Procedure `Neig`.
- Pick the solution whose sum coloring is the minimum among the four coloring solutions.

4.4 Analysis

In this subsection we analyze the approximation ratio of Algorithm $\mathcal{C}$. All through the analysis, let $Z = (n - I_1)/2 + \epsilon_d n$ and $W = (n - I_1)/2 - \epsilon_d n$. The term $\epsilon_d n$ quantifies the extent in which the graph induced by $Z \cup W$ is unbalanced. This is the graph resulting once the maximum independent set I_1 is deleted from G.

Outline of the analysis:

- If $Z - W = 2\epsilon_d n$ is "large" enough, then already $\min\{\texttt{SC(A(2))}, \texttt{SC(A(3))}\}$ yields the 10/9-ratio.
- Otherwise, $Z - W$ is not too "large". If I_2 is "large" enough, then this time already $\min\{\texttt{SC(A(2))}, \texttt{SC(A(4))}\}$ yields the 10/9-ratio.
- Otherwise, W is almost as "large" as Z and I_2 is not too "large". If W is "small" enough and therefore Z is also "small" and I_1 is "large" enough, then $\texttt{SC(A(3))}$ alone yields the 10/9-ratio.
- Otherwise $Z - W$ and I_2 are not too "large" and W is not too "small". If the optimal algorithm does not deviate much from algorithm `A(3)`, then again $\min\{\texttt{SC(A(2))}, \texttt{SC(A(3))}\}$ yields the 10/9-ratio.
- Finally, if all the previous conditions do not hold, we use the new procedure `Neig` and show that $\min\{\texttt{SC(A(2))}, \texttt{SC(Neig)}\}$ yields the 10/9-ratio.

The analysis is therefore partitioned into five cases. In each case, we make some assumption A, proving that under this assumption, the $10/9-$ratio is guaranteed. We, therefore, continue the analysis assuming that A does not hold.

Case 1: $\epsilon_d \geq 1/40$.

In this case, consider the performance of the best of the two algorithms `A(2)` and `A(3)`. Clearly $\texttt{SC}(\texttt{A(2)}) \leq 3n/2$ (see section 2.5). Algorithm `A(3)` colors I_1 by 1, Z by 2, and W by 3. Hence,

$$\begin{aligned} \texttt{SC}(\texttt{A(3)}) &= I_1 + 2Z + 3W \\ &= \frac{5n - 3I_1}{2} - \epsilon_d n \\ &\leq \frac{5n - 3I_1}{2} - n/40\ . \end{aligned}$$

On the other hand, the optimal coloring, `OPT`, colors at most I_1 vertices by 1 and the rest of the vertices by at least 2. This implies that $\texttt{SC}(\texttt{OPT}) \geq 2n - I_1$. It follows that $\texttt{SC}(\texttt{A(3)})/\texttt{SC}(\texttt{OPT})$ increases when I_1 decreases. Therefore, The worst case for algorithm $\mathcal{C}$ is when $\texttt{SC}(\texttt{A(2)}) = \texttt{SC}(\texttt{A(3)}) = 3n/2$. We get $3n/2 = (5n - 3I_1)/2 - n/40$, which implies that $I_1 = 13n/20$. For this value of I_1 the lower bound for $\texttt{SC}(\texttt{OPT})$ is $27n/20$. The 10/9 bound follows since

$$\frac{\texttt{SC}(\mathcal{C})}{\texttt{SC}(\texttt{OPT})} \leq \frac{3n/2}{27n/20} = \frac{10}{9}\ .$$

Hence, we may continue the analysis under the following assumption.

Assumption 1 $\epsilon_d \leq 1/40$. *Let* $\epsilon = 1/40 - \epsilon_d$.

Case 2: $I_2 \geq Z + \frac{W}{3} + \frac{2\epsilon}{3}n$.

We prove here, in a proof similar to case 1, that the best of Algorithms `A(2)` and `A(4)` always has a 10/9−ratio. Algorithm `A(4)` has the following properties. It colors I_1 by 1, I_2 by 2, at least half of the remaining $(n - I_1 - I_2)$ vertices by 3, and the rest of the vertices by 4. The worst case for `A(4)` is when I_2 is minimal, in this case it happens when $I_2 = Z + W/3 + (2\epsilon n)/3$. This gives the following upper bound:

$$\begin{aligned} \texttt{SC}(\texttt{A(4)}) &\leq I_1 + 2I_2 + 3\left(\frac{n - I_1 - I_2}{2}\right) + 4\left(\frac{n - I_1 - I_2}{2}\right) \\ &\leq I_1 + 2\left(Z + \frac{W}{3} + \frac{2\epsilon n}{3}\right) + 3\left(\frac{W}{3} - \frac{\epsilon n}{3}\right) + 4\left(\frac{W}{3} - \frac{\epsilon n}{3}\right) \\ &= I_1 + 2Z + 3W - \epsilon n \\ &= \frac{5n - 3I_1}{2} - (\epsilon + \epsilon_d)n \\ &= \frac{5n - 3I_1}{2} - \frac{n}{40}\ . \end{aligned}$$

The 10/9−ratio follows as in case 1, where again we choose the best between two algorithms one with $\texttt{SC} = (5n - 3I_1)/2 - n/40$ and one with $\texttt{SC} = 3n/2$.

Assumption 2 $I_2 \leq Z + \frac{W}{3} + \frac{2\epsilon}{3}n$.

Case 3: $W \leq (5\epsilon n + 6\epsilon_d)n$.

In this case we consider only the performance of Algorithm `A(3)`. Recall that $\mathtt{SC(A(3))} = I_1 + 2Z + 3W$, while $\mathtt{SC(OPT)} \geq 2n - I_1$. Therefore,

$$\begin{aligned}
\frac{\mathtt{SC(A(3))}}{\mathtt{SC(OPT)}} &\leq \frac{I_1 + 2Z + 3W}{2n - I_1} \\
&= 1 + \frac{W}{n + Z + W} \\
&= 1 + \frac{W}{n + 2W + 2\epsilon_d n} \\
&\leq 1 + \max_W \left\{ \frac{W}{n + 2W + 2\epsilon_d n} \right\} \\
&\leq 1 + \frac{5\epsilon + 6\epsilon_d}{1 + 10\epsilon + 14\epsilon_d} .
\end{aligned}$$

The last inequality follows since the expression increases with W, and in this case W is bounded by $(5\epsilon + 6\epsilon_d)n$. Since $\epsilon = (1/40) - \epsilon_d$, it follows that

$$\frac{\mathtt{SC(A(3))}}{\mathtt{SC(OPT)}} \leq 1 + \frac{1/8 + \epsilon_d}{1 + 1/4 + 4\epsilon_d} \leq 1 + \frac{6/40}{1 + 14/40} = \frac{10}{9}.$$

The last inequality follows since the expression increases with ϵ_d and $\epsilon_d \leq 1/40$ by Assumption 1.

Assumption 3 $W \geq (5\epsilon + 6\epsilon_d)n$.

Before dealing with the forth case, we need to prove some claims. The following corollary is derived directly from Assumptions 2 and 3.

Corollary 4.2 $Z + W - I_2 \geq (\frac{8\epsilon}{3} + 4\epsilon_d)n$. ▮

Note the implication of Corollary 4.2. Suppose that the number of vertices colored by 2 in `OPT` roughly equals I_2. Then since `OPT` cannot color more than I_1 vertices by 1, the corollary indicates that roughly $Z + W - I_2$ vertices in `OPT` are colored by at least 3. Taking this in mind, we make the following definitions. Let I_1^* be the set of vertices colored by 1 in `OPT`. Let A be the set of vertices from $Z \cup W$ that are colored by 1 in `OPT`. That is, $A = I_1^* \cap (Z \cup W)$. Note that the set $N_1(A) = N(A) \cap I_1$ is thus not colored by 1 in `OPT`. More precisely, the set of vertices not colored by 1 in `OPT`, denoted by ALT, equals, $(Z \cup W \cup N_1(A)) \setminus A$. The next claim states that the maximum independent set, $MIS(ALT)$, in the graph induced by ALT is "small enough". The corollary that follows the claim indicates that "many" of the vertices of ALT are colored by at least 3.

Claim 4.3 $MIS(ALT) = MIS((Z \cup W \cup N_1(A)) \setminus A) \leq Z + W - \left(\frac{8\epsilon}{3} + 4\epsilon_d\right) n + N_1(A)$.

Proof: We have: $MIS((Z \cup W \cup N_1(A)) \setminus A) \leq MIS((Z \cup W \cup N_1(A)) \leq MIS(Z \cup W) + N_1(A)$. Now, the desired inequality follows from Corollary 4.2, since $MIS(Z \cup W) = I_2$. ▮

Corollary 4.4 *In* `OPT`, *exactly* $I_1 - N_1(A) + A$ *vertices are colored by* 1 *and at least* $(8\epsilon/3 + 4\epsilon_d)n - A$ *vertices are colored by* 3.

Proof: The first part of the claim is by definition. For proving the second part of the corollary, note that the number of vertices outside I_1^* is $(Z + W + N_1(A) - A)$ and that the bound on the independent set of $G \setminus I_1^*$ is $(Z + W - (8\epsilon/3 + 4\epsilon_d)n + N_1(A))$. Subtracting these two expressions gives $(8\epsilon/3 + 4\epsilon_d)n - A$. ▮

Thus the following bound on the optimal sum is derived.

$$\begin{aligned} \texttt{SC(OPT)} &\geq (I_1 - N_1(A) + A) + 2\left(Z + W - \left(\frac{8\epsilon}{3} + 4\epsilon_d\right) n + N_1(A)\right) \\ &\quad +3\left(\left(\frac{8\epsilon}{3} + 4\epsilon_d\right) n - A\right) \\ &= 2n - I_1 - (2A - N_1(A)) + \left(\frac{8\epsilon}{3} + 4\epsilon_d\right) n \ . \end{aligned} \tag{1}$$

We are now ready to deal with the forth case.

Case 4: $2A - N_1(A) \leq (2\epsilon + 4\epsilon_d)n$.

Plugging this upper bound in inequality 1 gives,

$$\texttt{SC(OPT)} \geq 2n - I_1 + \frac{2\epsilon n}{3} \ . \tag{2}$$

Recall, that the usual lower bound on SC(OPT) is $2n - I_1$, and thus this lower bound is an improvement. As in case 1, we consider the performance of the best of the two algorithms `A(2)` and `A(3)`. It turns out that the ratio `A(3)` increases when I_1 decreases. Hence the worst case is when SC(A(2)) = SC(A(3)) = $3n/2$. We get $3n/2 = (5n - 3I_1)/2 - \epsilon_d n$, which implies that $I_1 = (2/3 - 2\epsilon_d/3)n$. By inequality 2 we get that

$$\texttt{SC(OPT)} \geq 2n - I_1 + \frac{2\epsilon n}{3} = \frac{4n}{3} + \frac{2(1/40)n}{3} = \frac{27n}{20} \ .$$

The 10/9 bound follows as in case 1.

Assumption 4 *$2A - N_1(A) \geq (2\epsilon + 4\epsilon_d)n$.*

We are now ready to complete the analysis of algorithm $\mathcal{C}$. The next case is the remaining case.

Case 5: We now show that under Assumptions 1, 2, 3, and 4 it is enough to consider the combined performance of procedure `Neig` and algorithm `A(2)`. Recall that $A = I_1^* \cap (Z \cup W)$ is the subset of $Z \cup W$ that is colored by 1 in OPT. Let $A_Z = Z \cap A$, $A_W = W \cap A$, $N_1(A_Z) = N(A_Z) \cap I_1$, and $N_1(A_W) = N(A_W) \cap I_1$. It follows from assumption 4 that either $2A_Z - N_1(A_Z) \geq (\epsilon + 2\epsilon_d)n$ or $2A_W - N_1(A_W) \geq (\epsilon + 2\epsilon_d)n$. Consequently, either d_{S_Z} or d_{S_W} is greater or equal to $(\epsilon + 2\epsilon_d)n$. This is because procedure `Neig` finds the sets which maximize expressions of the form $2S - N(S)$.

The worst case happens when only $d_{S_Z} \geq (\epsilon + 2\epsilon_d)n$. This is because otherwise the gain of procedure `Neig` over algorithm `A(3)` is larger (see proposition 4.1). Hence, by proposition 4.1,

$$\begin{aligned}
\texttt{SC(Neig)} &= \texttt{SC(A(3))} + (Z - W) - d_{S_Z} \\
&\leq \frac{5n - 3I_1}{2} - \epsilon_d n + 2\epsilon_d n - (\epsilon + 2\epsilon_d)n \\
&= \frac{5n - 3I_1}{2} - (\epsilon + \epsilon_d)n \\
&= \frac{5n - 3I_1}{2} - \frac{n}{40}\ .
\end{aligned}$$

Thus, as in Case 1, the best of Procedure `Neig` and Algorithm `A(2)` yields the 10/9−ratio. We have completed the proof of the following theorem.

Theorem 4.5 *Algorithm $\mathcal{C}$ is a polynomial* 10/9*-approximation algorithm for the* `MBCS` *problem.* ▮

5 Discussion and future work

We have proven that the `MBCS` is NP−hard, and furthermore, admits no polynomial time approximation scheme, unless $P = NP$. We have also given an improved 10/9−ratio approximation algorithm for `MBCS`.

Some open questions to be addresed in future work.

- Determine the best constant-ratio approximation for `MBCS`. For example, is the 10/9-ratio algorithm the best possible? A good direction to check is the following: is it possible to choose a better set S to be moved away from $Z \cup W$, and be recolored 1 in `Neig`? Note that we re-color by 1 a set S_W or S_Z that is entirely contained in a single side (left side or right side). It may be possible to gain by choosing to be colored 1 an (independent) set S containing vertices of both Z and W. Choosing such a good S seams a challenging task.
- Give good approximation algorithms for `MCS` on other families of graphs, such as interval graphs. An important contribution was done in that respect in [NSS94], where a ratio 2 algorithm was presented for `MCS` on interval graphs. Can this ratio be improved? Furthermore, no MAXSNP-hardness result is known for `MCS` on interval graphs.

References

[ALM$^+$92] S. Arora, C. Lund, R. Motwani, M Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. In *Proc. of the 33'rd IEEE Symp. on the Foundations of Computer Science*, pages 14–23, 1992.

[AS90] B. Awerbuch and M. Saks. A Dining Philosophers Algorithm with Polynomial Response Time. In *Proc. of the 31'st IEEE Symp. on the Foundation of Computer Science*, pages 65–74, 1990.

[BP92] J. Bar-Ilan and D. Peleg. Distributed Resource Allocation Algorithms. In *Proc. of the Sixth International Workshop on Distributed Algorithms*, pages 276–291, 1992.

[BST96] A. Bar-Noy, H. Shachnai, and T. Tamir. On chromatic sums and distributed resource allocation. In *Proc. of the fourth Israel Symp. on Theory and Computing and Systems*, pages 119–128, 1996.

[BBH$^+$96] A. Bar-Noy, M. Bellare, M. M. Halldórsson, H. Shachnai, and T. Tamir. On chromatic sums and distributed resource allocation. In URL: http://www.eng.tau.ac.il/ amotz/publications.html.

[BF94] P. Berman and M. Fuŕer. Approximating maximum independent set in bounded degree graphs. In *Proc. of the Fifth ACM-SIAM Symp. on Discrete Algorithms*, pages 365–371, 1994.

[BCC$^+$94] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan. The Minimum Latency Problem. In *Proc. of the 26'th IEEE Symp. on the Theory of Computing*, pages 163–171, 1994.

[CM84] K. Chandy and J. Misra. The Drinking Philosophers Problem. *ACM Trans. Programming Languages and Systems*, 6:632–646, 1984.

[Chr76] N. Christofides. Worst case analysis of a new heuristic for the traveling salesman problem. Technical report GSIA, Carnegie-Mellon Univ., 1976.

[E70] P. Erdös. On the Graph-Theorem of Turán. In *Math. Lapok*, 21:249–251, 1970.

[EKS] P. Erdös, E. Kubicka, and A. J. Schwenk. Graphs that Require Many Colors to Achieve their Chromatic Sum. *Congressus Numerantium*, 71:17–28, 1990.

[FK96] U. Feige and J. Kilian. Zero Knowledge and the Chromatic number. In *Proc. of the 11'th IEEE Conference on Computational Theory*, pages 278–287, 1996.

[GGT89] G. Gallo, M.D. Grigoriadis, and R.E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. on Comput.*, 18:30–55, 1989.

[Has96] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proc. of the 37'th IEEE Symp. on the Foundation of Computer Science*, 627–639, 1996.

[H83] D. S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics*, 6:243–254, 1983.

[HR93] M. M. Halldórsson and J. Radhakrishnan. Approximating the Chromatic Sum of a Graph. *Japan Advanced Institute of Science and Technology*, IS-RR-93-0002f, 1993.

[HR94] M. M. Halldórsson and J. Radhakrishnan. Greed is good: Approximating independent sets in sparse and bounded-degree graphs. In *Proc. of the 26'th IEEE Symp. on the Theory of Computing*, pages 439–448, 1994.

[K89] E. Kubicka. The Chromatic Sum of a Graph. PhD thesis, Western Michigan University, 1989.

[KKK89] E. Kubicka, G. Kubicki, and D. Kountanis. Approximation Algorithms for the Chromatic Sum. In *Proc. of the First Great Lakes Computer Science Conf.*, Springer LNCS 507, pages 15–21, 1989.

[KS89] E. Kubicka and A. J. Schwenk. An Introduction to Chromatic Sums. In *Proc. of the ACM Computer Science Conf.*, pages 39-45, 1989.

[L76] E. Lawler. Combinatorial Optimization Networks and Matroids. Holt Rinehart and Winston, 1976.

[LYN81] N. Lynch. Upper Bounds for Static Resource Allocation in a Distributed System. *J. of Computer and System Sciences*, 23:254–278, 1981.

[NSS94] S. Nicoloso and M. Sarrafzadeh and X. Song. On the sum coloring problem on interval graphs. *Unpublished Manuscript.*

[PY88] C. H. Papadimitriou and M. Yannakakis. Optimization approximation and complexity classes. In *Proc. of the 20'th IEEE Symp. on The Theory of Computing*, pages 229–234, 1988.

[T41] P. Turán. An Extremal Problem in Graph Theory. In *Mat. Fiz Lapok*, 48:436–452, 1941.

[TEA$^+$89] C. Thomassen, P. Erdös, Y. Alavi, j. Malde, and A. J. Schwenk. Tight Bounds on the Chromatic Sum of a Connected Graph. *J. of Graph Theory*, 13:353–357, 1989.